GSSC-0004

GAMMA-RAY LARGE AREA SPACE TELESCOPE (GLAST)

GLAST SCIENCE SUPPORT CENTER SOFTWARE MANAGEMENT PLAN

July 6, 2004

GODDARD SPACE FLIGHT CENTER
GREENBELT, MARYLAND

GAMMA-RAY LARGE AREA SPACE TELESCOPE
(GLAST)

GLAST SCIENCE SUPPORT CENTER
SOFTWARE MANAGEMENT PLAN

July 6, 2004

NASA Goddard Space Flight Center

Greenbelt, Maryland

DOCUMENT APPROVAL

Prepared by:

Robert Schaefer, GLAST Software Lead   Date

Concurrence:

Ken Lehtonen, GLAST Ground
System/Operations Manager   Date

Approved by:

Jay Norris, GLAST SSC Manager   Date

REVISION STATUS

This document is controlled by GSSC internal document configuration management.

Version	Date	Pages Affected	Description
Original	7/6/04	All	

Table of Contents

1. Introduction

1.1 Purpose

This management plan describes the activities, processes, and tools that the GLAST Science Support Center (GSSC) will use for the development of its operations, web, database access, and science analysis software. The GSSC is also helping the LAT team develop its Standard Analysis Environment, but management responsibility for these analysis tools will remain with the LAT team throughout the development phase of these tools. The GSSC has endeavored to keep its development environment similar to the LAT ground software environment, to satisfy the mission system specification that that there be a single ground system software development environment. (See GLAST Mission System Specification, 3.5.1.4). These environments will be the same in that, when C++ is used, we will follow the same coding standards, use the same compilers, the same configuration management tools, and the same code documentation conventions as defined by the LAT team.

1.2 Scope

This document covers all software developed by the GSSC. This document completely specifies the management plan for all such software except the portion of the Standard Analysis Environment software that the GSSC is helping to develop. For these latter tools, this document complements and augments the LAT documentation. However, this document does not replace or supersede any LAT document with regard to the Standard Analysis Environment. This document will be reviewed and updated throughout the project lifecycle to reflect the current software management plans.

1.3 Applicable Documents

The following documents contain information relevant to this plan:

- “GLAST Project Mission Operations Concept Document,” 433-OPS-0001
- “GLAST Project Ground System Requirements Document,” 433-RQMT-0006
- “GLAST Project Data Management Plan,” 433-PLAN-0009
- “GLAST Science Support Center Functional Requirements Document,” 433-RQMT-0002
- “GLAST Mission System Specification”, 433-SPEC-0001
- “GLAST Ground System Implementation Plan”
- “GSSC Development Plan” GSSC-0001
- “GSSC Design Document” GSSC-0003
- “GSSC Test Plan” GSSC-0005

The following NASA documents provide guidance for software development. They can be accessed online at http://swg.jpl.nasa.gov/resources/index.shtml.

NASA CM Gdbk	NASA Software Configuration Management Guidebook
NASA-GB-A201	NASA Software Assurance Guidebook
SMAP-GB-A301	NASA Software Quality Assurance Audits Guidebook
NASA-GB-A302	NASA Software Formal Inspections Guidebook
NASA-GB-001-94	NASA Software Measurement Guidebook
NASA-GB-001-95	NASA Software Process Improvement Guidebook
NASA-GB-001-96	NASA Software Management Guidebook
NASA-GB-001-97	Formal Methods Specification and Analysis Guidebook for the Verification of Software and Computer Systems, Volume II: A Practitioner's Companion
NASA-GB-002-95	Formal Methods Specification and Verification Guidebook for Software and Computer Systems, Volume I: Planning and Technology Insertion

1.4 Software Management Tools

The following tools will be used to maintain version control, build integrity, and in-line code documentation. We have also chosen to use the Eclipse integrated development environment to aid the software developer.

CVS – Concurrent Versioning System, http://www.cvshome.org/CVS/index_html
CMT – Code Management Tool, http://www.lal.in2p3.fr/technique/si/SI/CMT/CMT.htm
Doxygen – Automatic code documentation tool, http://www.doxygen.com
Eclipse – Unified platform for software development, http://www.eclipse.org/

1.5 GSSC Software Overview

The GSSC software is more fully described in the GSSC Design Document; here we just give an outline of functionality. The GSSC software includes the following components:

- Operations (Planning and Scheduling)
- User Interface
  - Proposal Preparation Tools
  - Web related software
- Data Archiving and Scientific Software
  - Database Tools
  - Data Pipelines
  - Internal Software tools

1.6 Terminology

- Software Module: a self-contained element of software. For configuration management purposes, this is the basic atom of software. This means that, organizationally, modules cannot be subdivided. Examples are: a perl module and a CMT package.
- Software Subsystem: a combination of two or more modules that function together in well defined contexts. A given module may be part of any number of subsystems.
- Software System: the union of all modules in a number of subsystems that are functionally related.
- GSSC Software System: the set of all software controlled by the GSSC.
- Standard Analysis Environment: a suite of tools for analyzing GLAST (primarily LAT) science data. The development of these tools is the responsibility of the LAT team.

2 Organization and Responsibilities

2.1 Organizational Responsibilities and Relationships

The GSSC Project is divided into functional units called sections, with each section manager reporting to the GSSC Manager (GM). Each section has section-specific software, however the bulk of software is contained in the Data Archiving and Software Support. This chapter describes the management structure of the software from all subsystems.

2.2 Software Manager

The GSSC Software Manager has overall responsibility for tracking and coordinating the development schedule, balancing resource allocation, and developing the software support infrastructure. The Software Manager must therefore serve on the Configuration Control Board to coordinate available resources for issue resolution assignment. In emergencies, the Software Manager can assign bug fixes to software personnel, with the understanding that the Configuration Control Board must eventually approve the resulting software. The Software Manager reports to the Data Archive and Software Support section manager, as the bulk of software development occurs in this section.

2.2.1 Software Configuration Manager

The Software Configuration Manager is responsible for the configuration management of the GSSC software. The LAT Science Analysis Software Configuration Manager is responsible for any software delivered by GSSC personnel. For purposes of this document, configuration management includes designing, implementing and distributing a coherent scheme for source code tracking, module building, code distribution, and adherence to coding and testing standards. The GSSC Configuration Manager reports to the GSSC Software Manager.

2.2.2 Software Test Manager

The Software Test Manager is responsible for managing the development and implementation of all software test procedures. The responsibilities of the Test Manager are described in the GSSC Test Plan.

2.3 Management Practices

The overall objective of software management is to deliver quality software that enables the GSSC ground operations and the SAE software to meet or exceed their performance and delivery requirements. This objective will be met through the application of:

- Cost and schedule management
- Risk management
- Configuration management
- Standard development, documentation and testing procedures
- Reviews
- A corrective action process

2.3.1 Schedule

The development schedule will be estimated and tracked as described in the GSSC Development Plan. The schedule for developing the Standard Analysis Environment will be coordinated with the LAT Science Analysis Software Manager at SLAC. Software development activities are managed by way of the Work Breakdown Structure (WBS). Work performance is assessed and reported to the GLAST Project Management monthly (in Project Status Reports) in order to track progress and identify when major corrective action may be required to minimize potential cost and schedule problems.

2.3.2 Configuration Management

Configuration management is an ongoing process through which software development and deployment processes are controlled throughout the life of the project. This process involves configuration identification, change control, status accounting, and configuration verification through code reviews.

2.3.2.1 Configuration Control Board

A group of software programmers and cognizant scientists (mainly from the GSSC) will form a configuration control board (CCB). The board will approve or reject proposed changes to the GSSC software, software development environment, coding standards, and documentation standards. The board will also prioritize and assign bug fixes, approve software change proposals, and approve new software versions for all GSSC software, including any and all LAT software whose maintenance is assumed by the GSSC. The board will consist of at least: the software manager, the three GSSC section managers, and the GSSC Manager.

The CCB will manage software only after it has been officially released during one of the GSSC releases.

2.3.2.2 Configuration Identification

Each code module will be assigned a version number that will serve as its configuration identification. The version number will be a three-field tag in which the first field is the major release number, the second field is the minor release number, and the third field is the patch number. A change in the major release number will indicate that major, backward incompatible changes or a significant number of enhancements to the code were made. A change in the minor release number will indicate that one or more new enhancements or features were added to the code. A change in the patch number will be used to indicate that only small changes were made to the code, such as bug fixes, that do not add enhancements or features. Once the code has been approved for a GSSC release, it will be given a cvs tag indicating the GSSC release number.

2.3.2.3 GSSC Software Releases

The GSSC will create unified releases of the GSSC software system. The schedule of these releases will be tied to the GLAST Ground System Ground Readiness Tests (GRTs). These releases are structured as follows. Each release will have a code freeze during which time tests will be carried out. The resulting bug-fixed code will carry a tag indicating the GSSC release number. These releases will be tracked in the GSSC software development schedule.

2.3.2.4 GSSC Software Freezes

Each GSSC software release will be preceded by a freeze of between one and two months in length. Software scheduled to be included in an upcoming release (see GSSC design document) will be version tagged and tested. Unit tests and subsystem tests will be carried out according to the GSSC Test Plan document. Only minor bug fixes will be allowed to the code during this time.

2.3.2.5 Change Control

Before a software tool has been tagged for an official release, changes can be made to the software at the discretion of the software manager in coordination with the section managers. Once code has been given a GSSC release tag, any changes must be approved by the GSSC CCB. Developers will follow the versioning standards for configuration identification at all times.

2.3.2.6 Configuration Status Accounting and Audits

All changes to controlled modules will be documented and tracked so that the status of change actions can be ascertained and verified. Periodic audits, run by the GSSC Test manager will confirm that approved changes have been implemented and tested in the released software.

2.3.2.7 Configuration Management Tools

The software development environment provided by the GSSC software group will incorporate CVS for source file management/tracking and CMT for compiled code software module management/building.

2.3.3 Software Development Standards

The GSSC software group will develop software development standards, with special regard to code modularity, testability, and maintainability. Standards will be published in a software development on-line manual and, where possible, enforced by the code management system.

2.3.4 Reviews

All software designs will be subjected to peer reviews and formal reviews. Peer reviews will be conducted at periodic intervals and will involve an informal presentation of code designs, specifications and implementations to scientists and programmers within the GSSC project. The Software Manager will schedule these reviews. Peer reviewers will provide the Software Manager with written reports that include recommendations for addressing weaknesses found in reviews. These reports will be archived and checked at future reviews and audits to ensure that code problems are being addressed. The GSSC software group will also participate in the formal Users' Committee reviews.

3 Software Development Activities

3.1 Software Life Cycle

The software development environment provided by the GSSC software group will support a formal software lifecycle, from design through maintenance of mature production modules.

3.2 Design and Development Tools

3.2.1 Development Platform

The GSSC will develop software on Intel x86-compatible hardware, using the Linux and/or Windows operating systems. The versions of Windows and Linux may evolve with time. These changes will be coordinated with the LAT Science Analysis Software group.

3.2.2 Target Platforms

All GSSC software will run on Intel x86-compatible hardware, using the Linux operating system. Distributed software will be supported on all platforms that are adopted for the SAE tools.

3.2.3 Programming Languages

The primary languages for GSSC software will be C++ and C. Scripting languages (Unix C shell/Bourne shell, Perl, Python, and Tcl/Tk) will be used as needed for web programs, data pipeline, and operations software, and to aid development (e.g., prototypes, code management, and automated tests). Scripts related to the SAE shall be in Python. All SAE software will be in C++, C or Python.

3.2.4 Development Environment

The primary development environment selected for the compiled C/C++ code is the CDT tool, which is part of the Eclipse platform. This tool was selected for platform independence, and its ability to interface seamlessly with both the CVS and CMT configuration tools in use. For scripting, developers are free to choose an editor and use the built-in debugging capabilities that come with the scripting compiler.

3.2.5 Documentation Tools

Doxygen will document code from comments embedded in the code itself when possible. Doxygen output will be posted to the web. Higher-level documents will be written with appropriate software tools (word processor, drawing program, etc.) then converted to standard formats as needed. Resulting documents will either be captured by the code management system or placed in the GLAST documentation repository.

3.2.6 Compilers

The GNU gcc compiler will be used for all compilation tasks on Unix and Linux; Microsoft's Visual Studio C++ compiler will be used on Windows platforms. The Eclipse platform is designed to be able to run both compilers.

3.3 Software Testing

Software testing will use a multi-tiered approach: tests include unit, subsystem and system tests. Detailed testing procedures are described in the GSSC Test Plan document.

3.3.1 Unit Testing

Each code module will provide a test program. Guidelines for rationalizing test construction and standard requirements for test programs' behavior will be determined by the Test Manager. The exact nature of any given module's test program will be determined by the complexity of the module it tests. Following any changes to source code, modules will be required to pass their unit element tests before the module can be accepted into production code.

3.3.2 Subsystem Tests

In this context the term "subsystem" refers to a combination of more than one code module. Test suites will be built up which exercise more than one code module together. The upgrade of a member module prompts the running of any test that includes that module.

3.3.3 System Tests

GLAST project-wide tests of the entire ground system will test the software as part of a larger ground system. These tests include Ground Readiness Tests (GRTs), End-To-End (ETE) tests, and GSSC System (pre-release) tests.

3.4 Documentation

GSSC software documentation will be tiered:

- GSSC Design Document
- Users' Guides
- Subsystem Design Documents
- Module/Tool Design documents
- Technical white papers
- Code package descriptive documents
- Code package reference documents

3.4.1 Design Documents

These include any documents that were used in the design of software modules and subsystems. Examples of these are use cases, pseudo-code, and architecture diagrams. These will be maintained on the GSSC internal web pages.

3.4.2 Users' Guides

Users' Guides will explain how to use the software, what it does, options available, and example uses. User guides will also be provided at the subsystem level, which give subsystem descriptions and give directions on how to do common tasks.

3.4.3 Module/Tool Design documents

These documents will be low level detail of considerations used in design, including GSSC internal software requirements documents, object diagrams, data flow diagrams, and logic diagrams. These will be kept on the GSSC internal website.

3.4.4 Technical White Papers

Technical white papers will range from proposals for GSSC software standards to detailed discussions of technical issues. These documents will be saved in the GSSC document repository.

3.4.5 Code Package Descriptive Documents

Code package descriptive documents consist of package-level manuals and coding examples. These documents evolve with the code and will be captured and tracked by the code management system.

3.4.6 Code Package Reference Documents

Code package reference documents will be generated from comments embedded in the source code and will thus be captured by the code management system. They will document both the package's external call interface and its internal structure. Such documents will be published to the web by the package building process.

3.5 Problem/Failure Reporting

The GSSC software group will track problems/failures through a web software-based issue tracking system (roundup) that will follow a problem from its identification to its resolution. Problems will be reviewed by the CCB and programmers assigned to perform fixes.

ACRONYM LIST

CCB	Configuration Control Board
CM	Configuration Management
ETE	End-to-End
GM	GSSC Manager
GRT	Ground Readiness Test
GSSC	GLAST Science Support Center
LAT	Large Area Telescope
SAE	Standard Analysis Environment
WBS	Work Breakdown Structure